

# Algoritmos y Estructuras de Datos II

## Segundo parcial

Sábado 30 de octubre de 2021

### Aclaraciones

- El parcial es individual en todas sus etapas, y a libro abierto.
- Se responderán consultas por Discord, sólo de interpretación del enunciado, a través del canal ‘Mesa de Docentes’ de la categoría PARCIALES.
- Cualquier aclaración sobre el enunciado se realizará a través del canal ‘anuncios-evaluaciones’.
- El parcial durará 4 horas, de 9 a 13hs, y la entrega se realizará a través del Campus hasta las 13:15hs, en forma estricta. Se desestimarán entregas fuera de tiempo, sin excepciones.
- El ejercicio se calificará con **Perfecto**, **Aprobado** (que puede ir con ‘-’ o ‘?’), **Regular**, o **Insuficiente**.
- Para aprobar la cursada se deben tener todos los ejercicios aprobados de todos los parciales.
- Los ejercicios con **R** o **I** se recuperarán por separado al final del cuatrimestre.
- Sólo se podrá tener un ejercicio con ‘A?’ en la cursada, si no, deberán recuperarse los que hagan falta.

### Ej. 1. Elección de estructuras

Se desea diseñar un sistema para manejar los teléfonos internos del nuevo edificio **Cero+Infinito**. De esta forma, haciendo honor a la gente de los departamentos e institutos de investigación que lo habitarán, los números de interno estarán todos en binario (i.e., serán secuencias de 0s y 1s) y podrán ser de cualquier longitud.

De forma de simplificar el funcionamiento de la central telefónica, se decidió que el conjunto de internos debe ser libre de prefijos. Esto es, que ningún interno puede ser prefijo de cualquier otro de los internos en la central. De esta forma, si ya existieran los internos 0010101 y 1111 no podrían agregarse el 001 ni el 0, así como tampoco el 11 o el 11110, entre otros, pero sí podrían agregarse los internos 10101, 0011 y 101, por nombrar sólo algunos.

Se quiere poder agregar internos a la central a medida que se asignen personas para trabajar en el nuevo edificio, además de poder eliminar aquellas extensiones que correspondan a gente que no trabaja más en el lugar. No puede haber usuarios de la central sin interno asignado.

A continuación se tiene una especificación para este problema.

**TAD DIGITO** es  $\{0, 1\}$

**TAD USUARIO** es NAT

**TAD CENTRALCEROÍNFINITA**

#### observadores básicos

interno : central  $c \times$  usuario  $u \longrightarrow \text{secu}(\text{dig})$   $\{u \in \text{usuarios}(c)\}$   
 usuarios : central  $\longrightarrow \text{conj}(\text{usuario})$

#### generadores

crearCentral :  $\longrightarrow \text{central}$   
 agregarInterno : central  $c \times \text{secu}(\text{dig}) \ i \times \text{usuario } u \longrightarrow \text{central}$   
 $\{u \notin \text{usuarios}(c) \wedge \forall i' : \text{secu}(\text{dig}) \text{ existeInterno?}(c, i') \Rightarrow \neg \text{esPrefijo}(i, i') \wedge \neg \text{esPrefijo}(i', i)\}$

#### otras operaciones

existeInterno? : central  $\times \text{secu}(\text{dig}) \longrightarrow \text{bool}$   
 empiezanCon : central  $\times \text{dig} \longrightarrow \text{central}$   
 vacia? : central  $\longrightarrow \text{bool}$   
 esPrefijo : secu(dig)  $\times \text{secu}(\text{dig}) \longrightarrow \text{bool}$   
 saturado? : central  $\longrightarrow \text{bool}$   
 eliminarInterno : central  $c \times \text{secu}(\text{dig}) \ i \longrightarrow \text{central}$   $\{\text{existeInterno?}(c, i)\}$

**axiomas**

```

interno(agregarInterno( $c, i', u'$ ),  $u$ )       $\equiv$  if  $u = u'$  then  $i'$  else interno( $c, u$ ) fi
usuarios(crearCentral)                      $\equiv$   $\emptyset$ 
usuarios(agregarInterno( $c, i, u$ ))           $\equiv$  Ag( $u$ , usuarios( $c$ ))
existeInterno?(crearCentral,  $i$ )            $\equiv$  false
existeInterno?(agregarInterno( $c, i', u$ ),  $i$ )  $\equiv$   $i = i' \vee$  existeInterno?( $c, i$ )
empiezanCon(crearCentral,  $d$ )               $\equiv$  crearCentral
empiezanCon(agregarInterno( $c, i, u$ ),  $d$ )     $\equiv$  if  $\neg$ vacía?( $i$ )  $\wedge_L$  prim( $i$ )= $d$  then
                                         agregarInterno(empiezanCon( $c, d$ ), fin( $i$ ),  $u$ )
                                         else
                                         empezarCon( $c, d$ )
                                         fi

vacía?(crearCentral)                        $\equiv$  true
vacía?(agregarInterno( $c, i, u$ ))             $\equiv$  false
esPrefijo( $s, s'$ )                          $\equiv$  vacía?( $s$ )  $\vee$  ( $\neg$  vacía?( $s'$ )  $\wedge_L$  prim( $s$ ) = prim( $s'$ )  $\wedge_L$ 
                                         esPrefijo(fin( $s$ ), fin( $s'$ )))
saturado?( $c$ )                              $\equiv$  if vacía?( $c$ ) then
                                         false
                                         else
                                         existeInterno?( $c, \langle \rangle$ )  $\vee$ 
                                         (saturado?(empiezanCon( $c, 0$ ))  $\wedge$  saturado?(empiezanCon( $c, 1$ )))
                                         fi
eliminarInterno(agregarInterno( $c, i', u$ ),  $i$ )  $\equiv$  if  $i = i'$  then
                                          $c$ 
                                         else
                                         agregarInterno(eliminarInterno( $c, i$ ),  $i', u$ )
                                         fi

```

**Fin TAD**

Dadas las siguientes operaciones, que son sólo algunas de las que posee el módulo, y de acuerdo a las complejidades temporales de peor caso indicadas, donde  $l$  es la longitud del interno a procesar y  $n$  es la cantidad de usuarios actuales de la central:

- AGREGARINTERNO(**inout** sistema: central, **in** interno: secuencia(digito), **in** usuario: idUsuario)  
 {**Pre**: El interno indicado no es prefijo de ningún otro de la central}  
 Agrega a la central el interno para el usuario indicado al sistema.  
Complejidad:  $O(l + \log(n))$
- ELIMINARINTERNO(**inout** sistema: central, **in** usuario: idUsuario)  
 {**Pre**: El interno del usuario indicado ya existe en el sistema.}  
 Elimina de la central el interno asociado al usuario indicado.  
Complejidad:  $O(\log(n))$
- EXISTEINTERNO?(**in** sistema: central, **in** interno: secuencia(digito))  $\longrightarrow$  res: bool  
 {**Pre**: Ninguna}  
 Indica si el interno informado está registrado en la central.  
Complejidad:  $O(l)$
- PUEDEAGREGARSE?(**in** sistema: central, **in** interno: secuencia(digito))  $\longrightarrow$  res: bool  
 {**Pre**: Ninguna}  
 Indica si el interno indicado podría agregarse a la central de acuerdo a las restricciones.  
Complejidad:  $O(l)$
- INTERNO(**in** sistema: central, **in** usuario: idUsuario)  $\longrightarrow$  interno: secuencia(digito)  
 {**Pre**: El usuario indicado está registrado en el sistema}  
 Obtiene el interno de un usuario determinado.  
Complejidad:  $O(\log(n))$

- **USUARIOS**(**in** sistema: central)  $\rightarrow$  usrs: conjunto(idUsuario)  
  {**Pre**: Ninguna}  
  Obtiene el conjunto de usuarios actuales de la central.  
  Complejidad:  $O(1)$

Se pide:

- a) Plantear la estructura de representación del módulo **CENTRALCEROINFINITA**, que provea las operaciones mencionadas más arriba. Se debe explicar qué información se guarda en cada parte, y las relaciones entre ellas.
- b) Resumir de qué manera se resolvería cada una de las operaciones listadas anteriormente, de acuerdo la estructura de representación elegida y la cota de complejidad solicitada, y haciendo las aclaraciones necesarias sobre *aliasing*.
- c) Escribir el algoritmo de las operaciones **AGREGARINTERNO** y **ELIMINARINTERNO** y justificar de manera formal y detallada su complejidad en peor y mejor caso.
- d) Escribir de manera coloquial y formal el invariante de representación.
- e) Escribir de manera formal la función de abstracción.
- f) ¿Qué habría que modificar en la estructura y en la implementación de las operaciones vistas, de ser necesario, si se deseara implementar la siguiente operación? Justificar.
  - **SATURADO?**(**in** sistema: central)  $\rightarrow$  res: bool  
  {**Pre**: Ninguna}  
  Indica si el sistema ya está saturado y no es posible agregar más internos sin romper la restricción.  
  Complejidad:  $O(1)$

Para la resolución del ejercicio no está permitido utilizar módulos implementados con tabla hash de base. Esto se debe a dos motivos: uno porque queremos que combinen el resto de las estructuras vistas, y otro porque los peores casos de la tabla hash exceden los que se piden en los ejercicios (por ejemplo, duplicar el espacio en una tabla cuesta  $O(n)$  en el peor caso).

Donde se menciona a **secuencia** o **conjunto** en las definiciones de las operaciones anteriores se debe usar alguno de los módulos vistos que se explican con **SECUENCIA** o **CONJUNTO**, respectivamente, de acuerdo a lo que está en el apunte de Módulos Básicos, o definir módulos nuevos que también se expliquen con esos TADs.